ESC-TR-2003-076

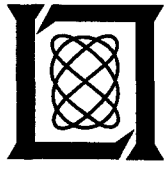
Project Report
PCA-IRT-3

Preliminary Design Review: GMTI Narrowband for the Basic PCA Integrated Radar-Tracker Application

A.I. Reuther

27 February 2003
Issued 6 February 2004

Lincoln Laboratory
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
LEXINGTON, MASSACHUSETTS



Prepared for the Defense Advanced Research Projects Agency
under Air Force Contract F19628-00-C-0002.

This report is based on studies performed at Lincoln Laboratory, a center for research operated by Massachusetts Institute of Technology. This work was sponsored by DARPA/ITO under Air Force Contract F19628-00-C-0002.

The ESC Public Affairs Office has reviewed this report, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This technical report has been reviewed and is approved for publication.

FOR THE COMMANDER

Gary Tutungian
Administrative Contracting Officer
Plans and Programs Directorate
Contracted Support Management

Non-Lincoln Recipients

PLEASE DO NOT RETURN

Permission is given to destroy this document when it is no longer needed.

Massachusetts Institue of Technology
Lincoln Laboratory

Preliminary Design Review: GMTI Narrowband for the Basic PCA Integrated Radar-Tracker Application

A.I. Reuther
Group 102

Project Report PCA-IRT-3

27 February 2003
Issued 6 February 2004

Lexington Massachusetts

ABSTRACT

This report describes ground moving target indicator (GMTI) processing to be done for the PCA (polymorphism computing architecture) integrated radar-tracker application. GMTI processing of the raw radar data is done to extract targets; target reports are passed on to the tracker part of the integrated radar-tracker application.

The GMTI processing described in this report is used to minimize the effects of interference patterns and ground clutter and to recognize and extract targets on the ground. The radar is presumed to be using a narrowband signal. For each data set, adaptive processing is used to null interference patterns and ground clutter noise. These operations incur a high computational cost. Adaptive processing requires the calculation of an adaptive filter (involving a QR decomposition and forward/backsliding) at run time for each data cube, while target parameter estimation incurs a high computation cost as more targets are detected in each data cube.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

1. INTRODUCTION

This report describes ground moving target indicator (GMTI) processing to be done for the PCA (polymorphism computing architecture) integrated radar-tracker application. GMTI processing of the raw radar data is done to extract targets; target reports are passed on to the tracker part of the integrated radar-tracker application.

The GMTI processing described in this report is used to minimize the effects of interference patterns and ground clutter and to recognize and extract targets on the ground. The radar is presumed to be using a narrowband signal.[1] For each data set, adaptive processing is used to null interference patterns and ground clutter noise. These operations incur a high computational cost. Adaptive processing requires the calculation of an adaptive filter (involving a QR decomposition and forward/backsliding) at run time for each data cube, while target parameter estimation incurs a high computation cost as more targets are detected in each data cube.

These radar signal processing technologies facilitate the extraction of targets from radar signal that include interference patterns and ground clutter noise—conditions in which radars usually cannot detect any targets. Depending on the choices of antenna and radar parameters, most targets should be detected consistently. However, some targets may not be detected in every detection interval because of the trade-offs that exist in choosing these parameters and the nonlinear effects of adaptive processing.

This report describes the key features and elements for a preliminary design review of the GMTI narrowband radar processing component of the integrated moving target indicator/tracker application. This report includes:

- a high-level description of the GMTI narrowband radar processing chain
- specifications of the input and output for each of the GMTI radar processing chain elements
- detailed descriptions of each of the GMTI radar processing chain elements

1.1 THE GMTI RADAR PROCESSING CHAIN

Before delving into the details of each of the elements of the processing chain, a description of the entire GMTI narrowband processing chain is in order. The processing chain is illustrated in Figure 1. In this figure, the small numbers in the upper right corner of the boxes refer to the section in which that stage is discussed in greater detail. The shaded box with bold labels is described in [1].

---

[1] A wideband signal is a signal in which the bandwidth is a significant fraction of the center frequency, while a narrowband signal is one in which the bandwidth is an insignificant fraction of the center frequency. The determination of what constitutes a significant fraction of the center frequency is application specific.

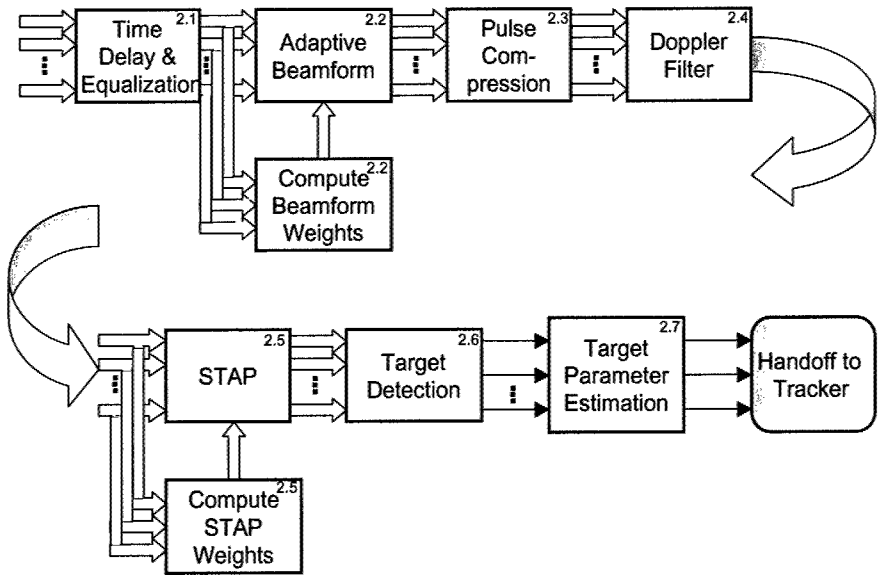


Figure 1. GMTI narrowband radar processing chain.

Before the radar data is processed by the GMTI stages, a radar signal consisting of a series of pulses comprising a coherent processing interval (CPI) has been transmitted. The pulse repetition interval (PRI) determines the time interval between transmitted pulses, and the number of pulses per CPI is given as N_{pri}. The signal reflects off of targets, the ground, and other entities and is received by the radar antenna. The antenna is of N_{fan} subarrays, each consisting of N_x horizontal by N_y vertical receiver elements. An array of N_{ch} sensor inputs, which are also called channels, is then derived from these receiver element inputs using weighted summations of radiation energy measurements. These channel sensor inputs are digitized by A/D converters at a rate of N_{rg} samples per PRI. The digitization rate alludes to the relationships between the range dimension and the PRI dimension: the range dimension is referred to as the fast time dimension, while the PRI dimension is the slow time dimension. This $N_{ch} \times N_{rg} \times N_{pri}$ digitized data cube of samples is the input to the GMTI processing chain.

The GMTI narrowband processing chain consists of seven stages: time delay and equalization; adaptive beamforming; pulse compression; Doppler filtering; space-time adaptive processing (STAP); detection; and estimation. This processing chain then reports the resulting targets to the tracker.

- The *time delay and equalization stage* compensates for differences in the transfer function between channel sensors.

- The *adaptive beamforming stage* transforms the filtered data into the beam-space domain to allow detection of target signals coming from a particular set of directions of interest while filtering out spatially-localized interference.

- The *pulse compression stage* filters the data to concentrate the signal energy of a relatively long transmitted radar pulse into a relatively short pulse response.
- The *Doppler filter stage* processes the data so that the radial velocity of targets relative to the platform can be determined.
- The *STAP stage* is a second beamforming stage which removes further interference and ground clutter interference.
- The *detection stage* uses constant false-alarm rate (CFAR) detection to compare a radar signal response to its surrounding signal responses to determine whether a target is present and uses target grouping to eliminate multiple target reports that are actually just one target.
- The *estimation stage* estimates target positions in order to pass them to the tracking algorithms.
- Finally, the target report is passed to the tracking algorithms which are described in [1].

1.2 GMTI CONTROL INPUT DATA

The GMTI processing function requires a control input which is coded as a Matlab structure. The names of the elements, the variables they correspond to in this report, and the descriptions of each element are given in Table 1. This structure must be passed to the GMTI function when it is called.

TABLE 1

Control Input Parameters for GMTI Processing

Parameter	Element Name	Description
N_{ch}	Nch	Number of radar sensor channels
N_{rg}	Nrg	Number of range gates per radar data cube
N_{pri}	Npri	Number of pulse repetition intervals per radar data cube
N_{tde}	Ntde	Number of FIR filter taps in the time delay and equalization filter
α_{ab_i}	alphaAbf	Diagonal loading factor in the adaptive beamforming stage
α_{stap}	alphaStap	Diagonal loading factor in the STAP stage
N_{pc}	Npc	Number of filter taps in the pulse compression FIR filter
N_{bm}	Nbm	Number of beams in the radar data cube after the adaptive beamforming stage
N_{dop}	Ndop	Number of Doppler bins for Doppler filtering
N_{stag}	Nstag	Number of PRI staggers produced by Doppler filtering for use in the STAP stage
N_{abfTS}	NabfTS	Number of training samples for the adaptive beamforming stage
N_{stapTS}	NstapTS	Number of training samples for the STAP stage
N_{cnb}	Ncnb	Number of clutter nulled beams in the radar data cube after the STAP stage
N_{fam}	Nfam	Number of receiver subarrays
N_x	Nx	Number of receiver elements in a subarray in the horizontal dimension
N_y	Ny	Number of receiver elements in a subarray in the vertical dimension
N_{cfar}	Ncfar	Number of range gates used in forming the noise estimate from each side of the cell under test
G	G	Number of guard range cells on both sides of the cell under test
μ	mu	Normalized target power threshold

The control input parameters presented in the table above are also shown in tables at the end of the stage section in which they are used.

Beyond the control input parameters, there are a number of data product inputs, as shown in Table 2.

TABLE 2
Data Product Inputs for GMTI Processing

Parameter	Element Name	Description
τ_{bm}	`Tbm`	Vector of range gate indices for adaptive beamforming stage training samples
h_{pc}	`hpc`	Coefficient vector of the pulse compression filter
τ_{stap}	`Tstap`	Vector of space-time snapshot indices for the training data for space-time adaptive processing

1.3 GMTI INPUT DATA

The input data to the GMTI radar processing chain is a series of three-dimensional radar data cubes whose dimensions are channel, range, and pulses (see Figure 2). The collection of the samples of the radar data cubes is explained in Section 1.1. Individual elements of the radar data cube C are referred to as $C(i, j, k)$, where i is the channel index, j is the range index, and k is the pulse index.

The series of radar data cubes will be indexed using a cell array in Matlab:[2]

```
input_data{1} = cube1;
input_data{2} = cube2;
input_data{3} = cube3;
etc.
```

Depending on the size of the data cubes, they can be generated separately and stored or they can be generated as they are needed.

[2] Items in `Courier font` correspond to functions or variables in the Matlab code.

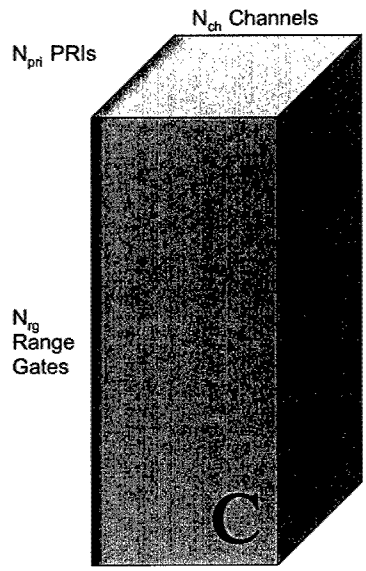


Figure 2. Radar data cube, the input data to the GMTI processing.

1.4 GMTI OUTPUT DATA

The tracker receives the following data for each target:

- target power or signal-to-noise ratio (SNR)
- target azimuth (in degrees or radians)
- target range (in meters or kilometers)
- target radial velocity (in m/s or km/h)
- target time stamp

The tracker is described in [1].

2. FUNCTIONAL DESCRIPTION

In this section, each of the processing stages are described in greater detail. For convenience, Figure 3 shows the GMTI narrowband processing chain with the sizes of the passed parameters.

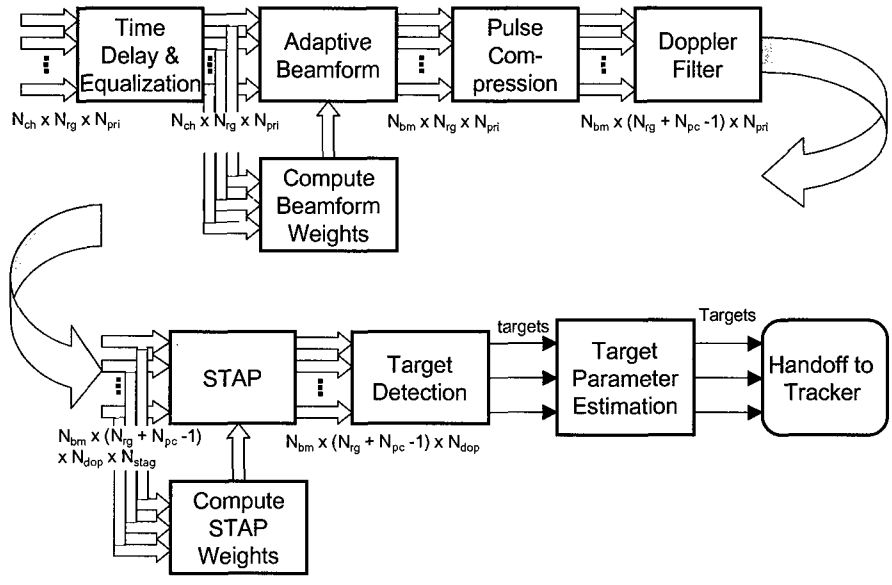


Figure 3. GMTI narrowband processing chain with passed parameter sizes.

2.1 TIME DELAY AND EQUALIZATION

Input: Data cubes of size $N_{ch} \times N_{rg} \times N_{pri}$.

Output: Data cubes of size $N_{ch} \times N_{rg} \times N_{pri}$.

The time delay and equalization stage is applied to each data band, and it uses a finite impulse response (FIR) filter on each range vector for each channel and each pulse to compensate for signal differences between channel sensors. That is, in Matlab notation, it operates on each $x = C(i, :, k, s)$, where i is the channel index and k is the PRI index. In its simplest form, a single tap adjustment can be folded into the digital beamforming weights; however, more complex equalization requires using programmable FIR filters. The FIR filter has filter coefficients, $h_{tde}[l]$, $0 \le l \le N_{tde} - 1$. The output of the filter, the vector y, is the convolution of the filter h_{tde} with the input x:

$$y[j] = \sum_{l=0}^{N_{tde}-1} x[j-l]h_{tde}[l], \text{ for } j = 0, 1, \ldots N_{rg} \quad . \tag{1}$$

Depending on the number of filter coefficients, N_{tde}, this filter can be implemented using fast convolution with FFTs [4] or by explicitly convolving the input signal and the filter coefficients as depicted in the equation above.

The time delay and equalization stage parameter is listed in Table 3.

TABLE 3

Time Delay and Equalization Stage Parameter

Name	Description
N_{tde}	Number of taps in the time delay and equalization filters

2.2 ADAPTIVE BEAMFORMING

Input: Data cubes of size $N_{ch} \times N_{rg} \times N_{pri}$.

Output: Data cubes of size $N_{bm} \times N_{rg} \times N_{pri}$.

The adaptive beamforming stage transforms the filtered data into the beam-space domain to allow detection of target signals coming from a particular set of directions of interest while filtering out jamming (spatially-localized) interference. The filtering is executed for each radar data cube within each CPI.

First, a steering matrix, $V1$, and antenna subarray response matrix, $T1$, must be determined for the relative direction from which the radar signal is being received. For this simulation, the steering matrix only involves the azimuth angle and ignores the elevation angle by assuming that it is set to zero. The vector of the steering azimuth angles is determined with a linear distribution over the beam range of the radar with endpoints at the –3 dB signal attenuation points of a radar beam. Assuming that the –3 dB signal attenuation points of a radar beam are at $\pm 1° = \pm\pi/180$ rad and the look angle of the radar is Θ, then the vector of the steering azimuth angle is $\Phi = \left[\Theta - \frac{\pi}{180} \ldots \Theta + \frac{\pi}{180}\right]$, such that the points in between are evenly spaced angles. Φ has a total of N_{bm} points.

$T1$ is the antenna subarray response matrix and is of dimension $(N_x \cdot N_{fam}) \times N_{ch}$, where N_x is the number of receiver elements (or receiver modules) in a subarray in horizontal direction, and N_{fam} are the number of horizontal-direction subarrays. To acquire the $T1$ matrix, one must process the antenna composition matrix, T, so that it represents the received energy of the look direction. The contents of the T matrix depend on the characteristics of the antenna; each channel is made up of a weighted summation of

the received energy at each element. Hence, the elements in each column of the matrix convey the weighting of the contribution element in the subarrays to that channel signal.

The T matrix must be processed to reflect both the look direction and the vector of steering azimuth angles. First, each element of T is processed for the look direction such that $T1_{m,n} = T_{m,n} e^{-j\pi(m\Theta)}$, where $m = 1, ..., (N_x \cdot N_{fam})$ and $n = 1, ..., N_{ch}$. Then defining h as a Hamming window vector of length $(N_x \cdot N_{fam})$, we can define $U_{i,k} = h_i \Phi_k e^{-j\pi(i\Theta)}$, where $i = 1, ..., (N_x \cdot N_{fam})$, $k = 1, ..., N_{bm}$, h_i is the ith element of h, and Φ_k is the kth element of Φ.

$V1'$ can now be computed as $V1' = (T1^H \cdot T1)^{-1} \cdot T(1^H \cdot U)$. Since taking the inverse of $(T1^H \cdot T1)$ can be numerically unstable, the method described in Appendix A should be employed. Defining $\hat{v}_k = T1^H \cdot U_k$, for $k = 1, ..., N_{bm}$, let $L' = lqhouse(T1^H)$; L' is a lower triangular $N_{ch} \times N_{ch}$ matrix. Now using t_k as a temporary vector of size N_{ch} by 1, we can solve $L' t_k = \hat{v}_k$ for t_k using forward-substitution and $(L')^H v'_k = t_k$ for v'_k using back-substitution. Then, $V1' = [v'_1 \ldots v'_k \ldots v'_{N_{bm}}]$. Finally, $V1$ is formed by taking the first N_{bm} columns of the resultant Q matrix of the QR decomposition of $V1'$, thereby making the columns of $V1$ orthogonal to each other.

To perform adaptive beamforming, a clutter-free data matrix X_{abf} is extracted from C. Typically, the first several range gates of a CPI are considered clutter-free so only the interference signals are sensed by the radar receiver. A total of N_{abfTS} column vectors are needed, and the choice of which range samples to use is given by the N_{abfTS}-length index vector $\tau_{bm} = \{\rho_1, ..., \rho_{NabfTS}\}$. Commonly, $N_{abfTS} = 5N_{ch}$. Let x_i be the ith column vector of X_{abf}. x'_i is formed by taking the $N_{ch} \times N_{pr}$ matrix at range gate ρ_i; in Matlab terms: $y_i = \left(\sum_{k=1}^{N_{pri}} C(:, \rho_i, k) \right)$, which becomes a column vector of length N_{ch}. Letting $\delta_k = \frac{1}{N_{pri}} e^{-(j \cdot 2\pi)kd}$ be the kth element of the N_{pri}-length column vector δ, where $k = 1, ..., N_{pri}$, and d is the normalized Doppler filter (a scalar number) that is opposite clutter which is dependent on look angle and system parameters. Also, let t be an N_{pri}-length Chebychev window filter column vector with a desired sidelobe attenuation, for instance, 60 dB. Now x'_i becomes the element-wise multiplication of y_i, δ, and t ($x_i = y_i .* \delta .* t$, in Matlab terms).

Now A_{abf} is calculated with X_{abf}, such that

$$A_{abf} = \left[\frac{1}{\sqrt{N_{abfTS}}} X_{abf} \sqrt{\alpha_{abf}} I \right], \tag{2}$$

and $\hat{R} = A_{abf}(A_{abf})^H$. Thus, A_{abf} is an $N_{ch} \times (N_{abfTS} + N_{ch})$ matrix and $\hat{R}$ is an $N_{ch} \times N_{ch}$ matrix. Note that a diagonal loading factor $\alpha_{abf} I$ is used to augment the diagonal of A_{abf} so that it has a lower probabiity of being singular.

To compute the columns of the beamforming, interference-nulling matrix, W_{bm} (of size $N_{ch} \times N_{ch}$), one must solve:

$$w_m = \frac{\hat{R}^{-1} v_m}{v_m^H (\hat{R})^{-1} v_m} \tag{3}$$

where w_m is the mth column of W_{bm}, $\hat{R}$ is the estimated covariance matrix, and v_m is the mth column of the steering matrix, $V1$, computed above. The denominator of Equation (3) is a scalar normalization factor. The above equation is referred to as a Wiener-Hopf equation for calculating adaptive filter weights.

While this explains the mathematics behind calculating the interference-nulling matrix, calculating the inverse of $\hat{R}$ is not used due to numerical stability issues. Instead LQ-decomposition and forward- and backward-substitution is used to calculate each of the w_m vectors. As explained in Appendix A, an LQ-decomposition is performed on A_{abf}, an $N_{ch} \times (N_{abfTS} + N_{ch})$ matrix: $L' = lqhouse(A_{abf})$ in Matlab; L' is a lower triangular $N_{ch} \times N_{ch}$ matrix. Now using t_m as a temporary vector of size $N_{ch} \times 1$, we can solve $L' t_m = v_m$ for t_m using forward-substitution and $(L')^H w'_m = t_m$ for w'_m using back-substitution. Finally, compute $w_m = w'_m / ((t_m)^H t_m)$ to take into account the denominator scalar normalization factor.

Refer to Appendix A for a deeper explanation of this method of solving this Wiener-Hopf equation.

Finally, W_{bm} is used to beamform the radar data cube. Using Matlab notation in defining the matrix $X_k = C(:, :, k)$ with size $N_{ch} \times N_{rg}$, the matrix Y_k as the output of the beamforming,

$$Y_k = (W_{bm})^H X_k \quad . \tag{4}$$

Thus, the matrix Y_k is of size $N_{bm} \times N_{rg}$, and $C'(:, :, k) = Y_k$. Figure 4 shows a conceptual data cube and its partitioning for this operation.

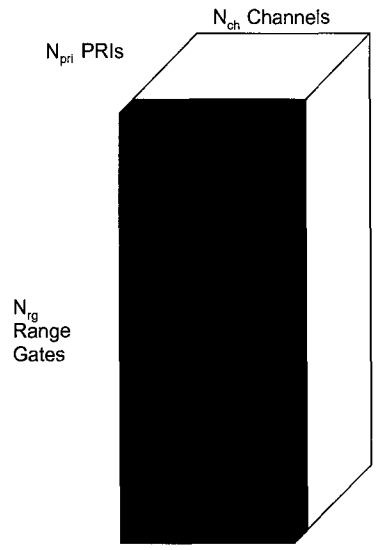


Figure 4. Data partitioning of input data cube for adaptive beamforming stage.

The data parameters of the adaptive beamforming stage are listed in Table 4.

TABLE 4
Adaptive Beamforming Stage Parameters

Name	Description
α_{abf}	Adaptive beamforming diagonal loading factor
N_{abfTS}	Number of training data set vectors in adaptive beamforming stage
τ_{bm}	Vector of range gate indices for adaptive beamforming stage training samples

2.3 PULSE COMPRESSION

Input: Data cubes of size $N_{bm} \times N_{rg} \times N_{pri}$.

Output: Data cubes of size $N_{bm} \times (N_{rg} + N_{pc-1}) \times N_{pri}$.

The pulse compression stage filters the data to concentrate the signal energy of a relatively long transmitted radar pulse into a relatively short pulse response. This compression is accomplished by applying a FIR filter operation to the range data for each pulse and channel within each radar data cube. That is, in Matlab notation, it operates on each x = *C(i, :, k)*, each of which are of length N_{rg}; this

partitioning is illustrated in Figure 5. The FIR filter has coefficients $h_{pc}[l]$, $l \in \{0 \ldots N_{pc}-1\}$. The output of the filter, the vector y of length $N_{rg}+N_{pc}-1$, is the convolution of the filter h_{pc} with the input x:

$$y[j] = \sum_{l=0}^{N_{pc}-1} x[j-l]h_{pc}[l], \text{ for } j = 0, 1, \ldots N_{rg} \quad . \tag{5}$$

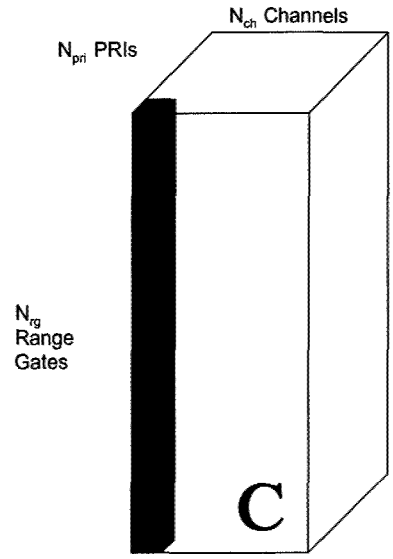


Figure 5. Data partitioning of input data cube for pulse compression stage.

Unlike the time delay and equalization stage, the number of taps in the pulse compression FIR filter is usually much larger than four or so taps. Therefore, it is usually more efficient to implement this convolution with the fast convolution technique using FFTs, possibly even considering an overlap-and-save method [4].

The pulse compression stage data parameter is listed in Table 5.

TABLE 5

Pulse Compression Stage Parameter

Name	Description
N_{pc}	Number of taps in the subband pulse compression filter

2.4 DOPPLER FILTERING

Input: Data cubes of size $N_{bm} \times (N_{rg} + N_{pc} - 1) \times N_{pri}$.

Output: Data cubes of size $N_{bm} \times (N_{rg} + N_{pc} - 1) \times N_{dop} \times N_{stag}$. In the Matlab implementation, the subband data cube is coded to be of size $(N_{bm} \cdot N_{stag}) \times (N_{rg} + N_{pc} - 1) \times N_{dop}$.

The Doppler filter stage processes the data so that the radial velocity of targets relative to the platform can be determined. Doppler filtering accomplishes this using a straightforward FFT applied to the pulse data for each range gate and beam. A further step made in the Doppler filtering which aids the STAP stage in filtering out ground clutter involves taking staggered pulse sample sets; N_{stag} is the number of pulse staggers. Using g ($1 \le g \le N_{stag}$) as the index of the stagger number, the staggers generated in Matlab notation are $y_{i,j,g} = C(i, j, g:(N_{pri} - N_{stag} + g))$; $y_{i,j,g}$ is a vector of length N_{pri}. For example, with $N_{stag} = 3$ the first input stagger vector would include the first pulse sample up to the $N_{pri} - 2$nd pulse sample, the second input stagger vector would include the second pulse sample up to the $N_{pri} - 1$ st pulse sample, and the third input stagger vector would include the third pulse sample up to the N_{pri}th (last) pulse sample. This is illustrated in Figure 6. With $N_{stag} = 1$, there is only one stagger which is the entire set of pulse samples for a given beam and range gate.

To smooth out the ringing effect of the FFT, a taper is applied to each $y_{i,j,g}$ vector. Let t be an N_{pri}-length Chebychev window filter column vector with a desired sidelobe attenuation, for instance, 60 dB. The tapered $y_{i,j,g}$ vector is an element-wise product: $y'_{i,j,g} = (y_{i,j,g})$.*t. The output of the Doppler filter is

$$z_{i,j,g} = FFT(y'_{i,j,g}) \quad , \tag{6}$$

for each beam, range gate, and stagger. Each $z_{i,j,g}$ is a vector of size N_{dop}. These vectors can be stored in a variety of ways; in the Matlab implementation, the Doppler filtered staggers are indexed using the channel/beam dimension. That is, the index of the first dimension of $C(i, j, k)$ accesses all of the beams of the first stagger ($g = 1$, $i = 1, ..., N_{bm}$) followed by all of the beams of the second stagger ($g = 2$, $i = N_{bm} + 1, ..., 2N_{bm}$) and so on.

The parameters of the Doppler processing stage are listed in Table 6.

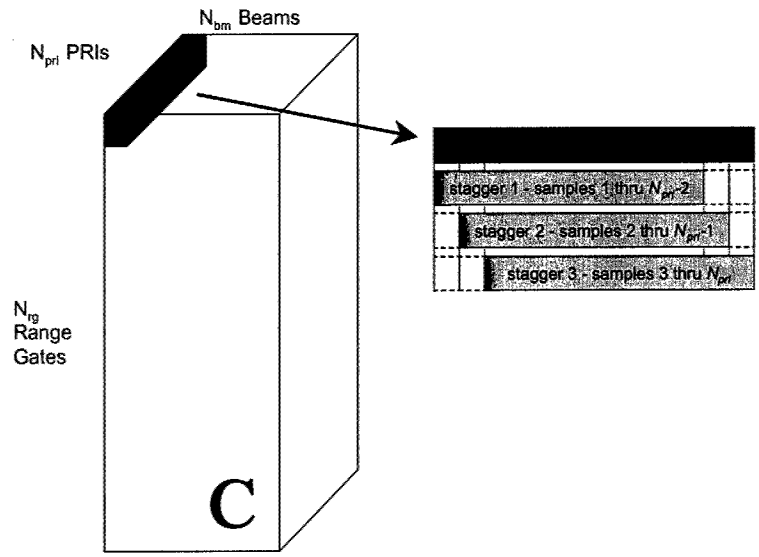


Figure 6. Data partitioning showing three staggers of input data cube for Doppler filtering stage.

TABLE 6
Doppler Processing Stage Parameters

Name	Description
N_{stag}	Number of pulse staggers

2.5 SPACE-TIME ADAPTIVE PROCESSING (STAP)

Input: Data cubes of size $N_{bm} \times (N_{rg} + N_{pc} - 1) \times N_{dop} \times N_{stag}$. In the Matlab implementation, the subband data cube is coded to be of size $(N_{bm} \cdot N_{stag}) \times (N_{rg} + N_{pc} - 1) \times N_{dop}$.

Output: Data cubes of size $N_{cnb} \times (N_{rg} + N_{pc} - 1) \times N_{dop}$.

The STAP stage is a second beamforming stage which removes further radar spatially-localized interference and ground clutter (space- and time-distributed) interference. Using the $z_{i,j,g}$ outputs from the Doppler filtering stage, define

$$x_{j,k,g} = \begin{bmatrix} z_{0,j,g}(k) \\ \cdots \\ z_{N_{bm}-1,j,g}(k) \end{bmatrix} . \tag{7}$$

In other words, $x_{j,k,g}$ is constituted with the N_{bm} complex elements for a given stagger, range gate, and Doppler gate. The size of $x_{j,k,g}$ is $N_{bm} \times 1$, and it is one stagger of samples. Next, the staggers are stacked upon each other to get a space-time snapshot:

$$x_{j,k} = \begin{bmatrix} x_{j,k,1} \\ \cdots \\ x_{j,k,N_{stag}+1} \end{bmatrix} \tag{8}$$

for the test cells, which makes $x_{j,k}$ an $N_{bm}N_{stag} \times 1$ vector. We will use these vectors of size $N_{bm}N_{stag} \times 1$ as our input data to the STAP filter, but we shall first build STAP steering vectors, and then compute the adaptive filter matrix.

As in the adaptive beamforming stage, a steering matrix, $V2$, and antenna subarray response matrix, $T2$, must be determined for the relative direction from which the radar signal has been received. $T2$ is calculated from the matrix multiplication of $T1$ and W_{bm}: $T2 = T1 \cdot W_{bm} \cdot T2$ is of dimensions $(N_x \cdot N_{fam}) \times N_{bm}$, where N_x is the number of receiver elements in a subarray in horizontal direction, and N_{fam} are the number of horizontal-direction subarrays.

Also as in the adaptive beamforming stage, the vector of steering azimuth angles is determined with a linear distribution over the beam range of the radar with endpoints at the –3 dB signal attenuation points of a radar beam. Assuming that the –3 dB signal attenuation points of a radar beam are at $\pm 1° = \pm\pi/180$ rad, and the look angle of the radar is Θ, then the vector of steering azimuth angles is $\Phi = \left[\Theta - \frac{\pi}{180} \ldots \Theta + \frac{\pi}{180}\right]$, such that the points in between are evenly spaced angles. But unlike the adaptive beamforming stage, Φ has a total of N_{cnb} points, instead of N_{bm} points. Then, as in the adaptive beamforming stage, defining h as a Hamming window vector of length $(N_x \cdot N_{fam})$, we can define $U_{i,k} = h_i \Phi_k e^{-j\pi(i\Theta)}$, where $i = 1, \ldots, (N_x \cdot N_{fam})$, $k = 1, \ldots, N_{cnb}$, h_i is the ith element of h, and Φ_k is the kth element of Φ.

$V2'$ can now be computed as

$$V2' = (T2^H \cdot T2)^{-1} \cdot (T2^H \cdot U) \quad . \tag{9}$$

Since taking the inverse of $(T2^H \cdot T2)$ can be numerically unstable, the method described in Appendix A should be employed. Defining $\hat{v}_k = T2^H \cdot U_k$, for $k = 1, \ldots, N_{cnb}$, let $L' = lqhouse(T2^H)$; L' is a lower triangular $N_{cnb} \times N_{bm}$ matrix. Now using t_k as a temporary vector of size $N_{ch} \times 1$, we can solve $L' t_k = \hat{v}_k$ for t_k using forward-substitution and $(L')^H v'_k = t_k$ for v'_k using back-substitution. Then, $V2' = [v'_1 \ldots v'_k \ldots v'_{N_{bm}}]$.

To this point, all of the processing applies to the subband data cube as a whole, but the rest of the processing must be completed for each Doppler bin, δ. Within each Doppler bin to accommodate the STAP calculation for each stagger, $V2'$ is replicated with an added phasor term which is dependent on the Doppler bin, δ. Letting the phasor equal:

$$\phi(g) = \exp\left(g \cdot \left(\left(\frac{(\delta-1)(N_{dop}/2)}{N_{dop}}\right) mod 1\right)\right) \quad , \tag{10}$$

where $g = 1, ..., N_{stag}$, $V2''$ is calculated as:

$$V2'' = [\phi(1) \cdot V2' \ldots \phi(g) \cdot V2' \ldots \phi(N_{stag}) \cdot V2'] \ . \tag{11}$$

$V2''$ is a $(N_{bm} \cdot N_{stag}) \times N_{cnb}$ matrix. Finally, $V2$ is formed by taking the first N_{cnb} columns of the resultant Q matrix of the QR decomposition of $V2''$, thereby making the columns of $V2$ orthogonal to each other. $V2$ is a $(N_{bm} \cdot N_{stag}) \times N_{cnb}$ matrix. The first N_{cnb} rows of the R matrix of the QR decomposition of $V2'$ are also retained and renamed $R2$, which is a $(N_{bm} \cdot N_{stag}) \times N_{cnb}$ matrix.

For each Doppler bin, a sample data matrix $X_{stap}^{(\delta)}$ is extracted from the output of the Doppler filtering stage. A total of N_{stapTS} column vectors are needed, and the choice of which range samples to use is given by the N_{stapTS}-length index vector $\tau_{stap} = \{\rho_1, \ldots, \rho_{N_{stapTS}}\}$. Commonly, $N_{stapTS} = 5N_{bm}N_{stag}$. Using the $x_{j,k}$ vectors of size $N_{bm}N_{stag} \times 1$, which presented at the beginning of this section

$$X_{stap}^{(\delta)} = \left[x_{\rho_1}, \delta \ldots \left(x_{\rho_{N_{stap}}} \delta\right)\right] . \tag{12}$$

Now $A_{stap}^{(\delta)}$ is calculated with $X_{stap}^{(\delta)}$, such that

$$A_{stap}^{(\delta)} = \left[\frac{1}{\sqrt{N_{stapTS}}} X_{stapTS}^{(\delta)} \sqrt{\alpha_{stap} I}\right] \ , \tag{13}$$

and $\hat{R}^{(\delta)} = A_{stap}^{(\delta)} (A_{stap}^{(\delta)})^H$. Thus, $A_{stap}^{(\delta)}$ is an $N_{bm}N_{stag} \times (N_{stapTS} + N_{bm}N_{stag})$ matrix, and $\hat{R}^{(\delta)}$ is an $N_{bm}N_{stag} \times N_{bm}N_{stag}$ matrix. Note that a diagonal loading factor $\alpha_{stap} I$ is used to augment the diagonal of $A_{stap}^{(\delta)}$ so that it has a lower probability of being singular.

To compute the columns of the beamforming, interference-nulling matrix, $W_{stap}^{(\delta)}$ (of size $N_{bm}N_{stag} \times N_{cnb}$), one must solve:

$$w_{\delta, b} = \frac{(\hat{R}^{(\delta)})^{-1} v_b}{v_b^H (R^{(\delta)})^{-1} v_b} \ . \tag{14}$$

where $w_{\delta,b}$ is the bth column of $w_{stap}^{(\delta)}$, $\hat{R}^{(\delta)}$ is the estimated covariance matrix, and v_b is the bth column (b=1, ..., N_{cnb}) of the steering matrix, $V2$, computed above. The denominator of Equation (14) is a scalar normalization factor. The above equation is referred to as a Wiener-Hopf equation for calculating adaptive filter weights.

While this explains the mathematics behind calculating the clutter-nulling matrix, calculating the inverse of $\hat{R}^{(\delta)}$ is not used due to numerical stability issues. Instead, LQ-decomposition and forward- and back-substitution is used to calculate each of the $w_{\delta,b}$ vectors (b = 1, ..., N_{cnb}) of the $W_{stap}^{(\delta)}$ matrix. As explained in Appendix A, an LQ-decomposition is performed on $A_{stap}^{(\delta)}$, an $N_{bm}N_{stag} \times (N_{stapTS} + N_{bm}N_{stag})$ matrix: $L_{\delta,b}$ = lqhouse($A_{stap}^{(\delta)}$; $L_{\delta,b}$ is a lower triangular $N_{bm}N_{stag} \times N_{bm}N_{stag}$ matrix. Now using t_b as a temporary vector of size $N_{bm}N_{stag} \times 1$, we can solve $L_{\delta,b}t_b = v_b$ for t_b using forward-substitution and $(L_{\delta,b})^H w'_{\delta,b} = t_b$ for $w'_{\delta,b}$ using back-substitution.

Lastly, compute $w_{\delta,b} = w'_{\delta,b}/((t_b)^H t_b)$ to take into account the denominator scalar normalization factor.

Refer to Appendix A for a more detailed explanation of this method of solving the Wiener-Hopf equation.

Finally, the adaptive weights vectors are applied to the snapshots $x_{j,\delta}$ of size $N_{bm}N_{stag} \times 1$:

$$y_{j,\delta} = (R2)^H (W_{stap}^{(\delta)})^H x_{j,\delta} \quad . \tag{15}$$

The resulting vector $y_{j,\delta}$ is of size $N_{cnb} \times 1$ and must be calculated for each $j = 1 \ldots N_{rg}$ and $\delta = 1 \ldots N_{dop}$.

The STAP stage data parameters are listed in Table 7.

TABLE 7

Space-Time Adaptive Processing Stage Parameters

Name	Description
N_{stag}	Number of pulse staggers
N_{stapTS}	Number of training data set vectors in the STAP stage
τ_{stap}	Vector of space-time snapshot indices for the training data set for space-time adaptive processing

2.6 TARGET DETECTION

Input: Processed radar data cube of size $N_{cnb} \times (N_{rg} + N_{pc} - 1) \times N_{dop}$.

Output: Coordinates of detected targets in radar data cube.

Though there are many techniques for detecting targets in the processed radar data cube including various integrators and hard limiters, this application uses a version of constant false alarm rate (CFAR) detection with three-dimensional grouping. The data parameters of the target detection stage are listed in Table 8.

TABLE 8

Target Detection Stage Parameters

Name	Description
N_{cfar}	Number of range gates used in forming the noise estimate from each side of the cell under test
G	Number of guard cells on both sides of the cell under test
μ	Normalized target power threshold

2.6.1 CFAR Detection

During CFAR detection, a local noise estimate is computed from the $2N_{cfar}$ range gates near the cell $C(i, j, k)$ under test. A number of guard gates G immediately next to the cell under test will not be included in the local noise estimate. (This number does not affect the throughput.) The range cells involved in calculating the noise estimate for a particular Doppler bin and beam are shown in Figure 7. For each cell $C(i, j, k)$, the value of the noise estimate $T(i, j, k)$ is calculated as:

$$T(i,j,k) = \frac{1}{2N_{cfar}} \sum_{l=G+1}^{G+N_{cfar}} |C(i,j+l,k)|^2 + |C(i,j-l,k)|^2 \quad . \tag{16}$$

At the boundaries when $j - (G + N_{cfar} - 1) < 1$ or $j + (G + N_{cfar}) > N$, only the side range gate cells that are within the data cube are included, and the denominator of the leading scaling factor is decreased to reflect the number of side range gate cells that were actually used for the calculation. For each cell $C(i, j, k)$, the quantity $|C(i,j,k)|^2 / T(i,j,k)$ is calculated; this represents the normalized power in the cell under test. If this normalized power exceeds a threshold μ, the cell is considered to contain a target. This threshold can be adaptively determined, but for this application, it is set to a constant. The only cells in the data cube that are processed by the following stages are those where a target has been detected.

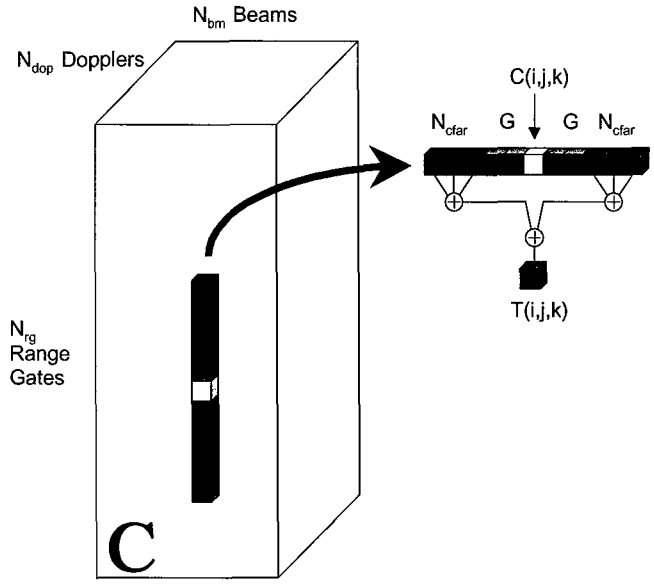


Figure 7. Data partitioning of input for CFAR detection stage.

2.6.2 Three-dimensional Grouping

During this step, detection in adjacent cells are grouped to avoid having multiple detections associated with the same target. The power of each target detected by the CFAR algorithm will be compared to the power of each cell in the $3 \times 3 \times 3$ cube centered on that cell. Therefore, each cell under test will be compared to 26 other cells ($27 - 1$, where $27 = 3^3$ and 1 cell is the cell under test, the cell in the middle of the cube). If the detection's raw power is the local maximum, it is retained. Otherwise, the detection is discarded; it is considered grouped with the detection in the cell with the higher power.

2.7 TARGET PARAMETER ESTIMATION

Input: Coordinates of detected targets in radar data cube.

Output: A target report for each detected target with sub-bin resolution.

This stage of the GMTI processing packages the target parameters for the target report. As presented in Section 1.4, a target report consists of the following data for each target:

- target power or signal-to-noise ratio (SNR)
- target azimuth (in degrees or radians)
- target range (in meters or kilometers)

- target radial velocity (in m/s or km/h)
- target time stamp

The target power is the quantity $|C(i,j,k)|^2 / T(i,j,k)$ for each target cell $C(i, j, k)$ which was calculated by the CFAR detection. The target time stamp will be reported as the input time stamp of the data cube from which this target was determined.

For the target azimuth, range, and radial velocity, sub-bin estimates can be determined by interpolating between the processed signal return points near the target. These interpolation estimators require a vector of data between which these interpolations will be calculated. A vector of interpolation points over the vector span are determined, and from these interpolation points, the sub-bin target location is where the maximum value is found in this interpolation vector. For this simulation software, the target azimuth angle estimate is conducted with a maximum likelihood estimator, while the target range and radial velocity (Doppler) estimates are conducted with a cubic spline interpolation estimator. The following two sections describe these two estimators. Finally, these sub-bin estimates are converted into their corresponding target data values.

2.7.1 Maximum Likelihood Estimator

To perform maximum likelihood estimation (MLE) of the azimuth angle, post-Doppler filtering, pre-STAP data must be used. Two hundred evenly-spaced azimuth angle samples are taken across the entire azimuth range, and the index of the angle at which the maximum value occurs is returned as the sub-bin estimate. All of these calculations must be made for each target report generated by the detection range.

The MLE function takes as input a vector of azimuth values, $x_{j,k}$, an $N_{bm}N_{stag} \times 1$ vector as calculated in Equation (8), where the target in question has been detected in range bin j and Doppler bin k. It also takes a vector of indices (azimuth bin numbers $[1{:}N_{bm}N_{stag}]$) which correspond to the vector of azimuth angles. (They must be of the same length.)

Similar to the STAP stage, a steering matrix, $V2$, and antenna subarray response matrix, $T2$, must be determined for the relative direction from which the radar signal is being received. $T2$ is calculated from the matrix multiplication of $T1$ and W_{bm}: $T2 = T1 \cdot W_{bm} \cdot T2$ is of dimensions $(N_x \cdot N_{fam}) \times N_{bm}$, where N_x is the number of receiver elements in a subarray in horizontal direction, and N_{fam} are the number of horizontal-direction subarrays.

Also similar to the STAP stage, the vector of steering azimuth angles is determined with a linear distribution over the beam range of the radar with endpoints at the –3 dB signal attenuation points of a radar beam. Assuming that the –3 dB signal attenuation points of a radar beam are at $\pm 1° = \pm\pi/180$ rad, and the look angle of the radar is Θ, then the vector of steering azimuth angles is $\Phi = \left[\Theta - \frac{\pi}{180} \ldots \Theta + \frac{\pi}{180}\right]$, such that the points in between are evenly spaced angles. But unlike the STAP stage, Φ has a total of 200 points, instead of N_{cnb} points. Then, as in the STAP stage, defining h as a Hamming window vector of

length $(N_x \cdot N_{fam})$, we can define $U_{i,k} = h_i \Phi_k e^{-j\pi(i\Theta)}$, where $i = 1, ..., (N_x \cdot N_{fam})$, $k = 1, ..., 200$, h_i is the ith element of h, and Φ_k is the kth element of Φ.

$V2'$ can now be computed as

$$V2' = (T2^H \cdot T2)^{-1} \cdot (T2^H \cdot U) \quad . \tag{17}$$

Since taking the inverse of $(T2^H \cdot T2)$ can be numerically unstable, the method described in Appendix A should be employed. Defining $\hat{v}_k = T2^H \cdot U_k$, for $k = 1, ..., N_{cnb}$, let $L' = lqhouse(T2^H)$; L' is a lower triangular $200 \times N_{bm}$ matrix. Now using t_k as a temporary vector of size $N_{ch} \times 1$, we can solve $L' t_k = \hat{v}_k$ for t_k using forward-substitution and $(L')^H v'_k = t_k$ for v'_k using back-substitution. Then, $V2' = [v'_1 ... v'_k ... v'_{N_{bm}}]$.

Since the target detection occurred in a specific Doppler bin, these calculations must only be conducted for that specific Doppler bin, δ. Within the calculations for that Doppler bin, $V2'$ is replicated with an added phasor term which is dependent on the Doppler bin, δ. Letting the phasor equal:

$$\phi(g) = \exp\left(g \cdot \left(\left(\frac{(\delta - 1)(N_{dop}/2)}{N_{dop}}\right) mod 1\right)\right) \quad , \tag{18}$$

where $g = 1, ..., N_{stag}$, $V2$ is calculated as:

$$V2 = [\phi(1) \cdot V2' ... \phi(g) \cdot V2' ... \phi(N_{stag}) \cdot V2'] \quad . \tag{19}$$

$V2$ is a $(N_{bm} \cdot N_{stag}) \times 200$ matrix. Using $A_{stap}^{(k)}$, the A_{stap} matrix for Doppler k calculated in Equation (13), we can find W_θ by solving the Wiener-Hopf equation: $A_{stap}^{(k)} W_\theta = V2$ for W_θ. Refer to Appendix A and paragraphs above to formulate this in a computationally stable manner.

W_θ is then used in the MLE equation to find a vector of sub-bin azimuth angle return estimates, Θ, a 200 element vector:

$$\Theta = \frac{\left|(W_\theta)^H x_{j,k}\right|^2}{(W_\theta)^H V2} \quad . \tag{20}$$

The maximum value location across the 200 elements of Θ must then be found and the corresponding sub-bin index value is returned.

2.7.2 Cubic Spline Interpolation Estimator

The cubic spline interpolation estimator, used to determine sub-bin range and Doppler estimates, takes as input a vector of values over which to estimate and a vector indices (bin numbers) which correspond to the vector of values. (They must be of the same length.) These vectors are assembled by taking each target bin position and including several points to each side of the target bin position within the dimension to be determined. So if a sub-bin range estimate is desired, and if the detection stage has found a target in azimuth bin i, range bin ρ, and Doppler bin k, and we are including three range points to either side of the target, the vector of values would be $x = C(i, \rho - 3{:}\rho + 3, k)$, in Matlab terms, and the index vector would contain $[\rho - 3{:}\rho + 3]$.

Cubic spline interpolation involves interpolating with a cubic equation between each set of adjacent points. The technique constrains the interpolation intervals to have continuous second derivatives which makes the complete interpolation curve over all of the value points smooth. The second derivatives can be computed for each of the points in the values vector, and the resulting system of equations which solve for the coefficients of other interpolation intervals can be assembled as a tridiagonal matrix solution. For a complete explanation of cubic spline interpolation, see Section 3.3 of [5].

To evaluation sub-bin estimates, one must construct a vector of sub-bin indices; for instance, we can subdivide the interval of indices into a total of 100 evenly-spaced points for which we want interpolated values. With the cubic interpolation equation coefficients of each interval, we can then evaluate the interpolated values at each of the 100 points, find the maximum value location, and return the corresponding sub-bin index value. All of these calculations must be made for each target report generated by the detection stage.

3. MATLAB IMPLEMENTATION

The GMTI processing chain will be initialized with a function with the following signature:

```
Gmti_param = GMTI_Init();
```

This function will load control parameters outlined in Section 1.2 into the structure `GMTI_params`.

The GMTI processing chain will be invoked with a function with the following signature:

```
targets = GMTI_run(input_data, GMTI_params);
```

This function call passes the aforementioned `GMTI_params` and the `input_data` outlined in Section 1.3, and it returns the target structure outlined in Section 1.4.

In order to keep memory usage to a reasonable level, the `GMTI` function will process only one radar data cube at a time, and within that radar data cube, it will process only one subband radar data cube at a time. Each of the stages depicted in Figure 1 will have its own function call within the `GMTI` function.

APPENDEX A
SOLVING THE WIENER-HOPF EQUATION

The Wiener-Hopf equation is used in both the adaptive beamforming and STAP stages. The equation $w = \tilde{R}^{-1} v$ defines an adaptive filter vector, w, as the product of the inverted estimated noise covariance matrix, $\tilde{R}$, and the nonadaptive steering vector, v. $\tilde{R}$ is composed using a set of training vectors, x_T, such that $\tilde{R} = (x_T)(x_T)^H$.

However, $\tilde{R}$ and its inverse are seldomly used to solve the Wiener-Hopf equation [3] because calculating $\tilde{R}$ and its inverse causes numerical instability. Instead, LQ-decomposition with forward- and back-substitution is used. First, the more general solution of the STAP Wiener-Hopf equation is discussed, and then the adaptive beamforming solution (in which a diagonal loading factor is added) is shown.

Compute the LQ decomposition $x_T = LQ$, where L becomes a lower triangular matrix and Q becomes a unitary matrix. The LQ decomposition is related to the QR decomposition described in [2], by inputting the Hermitian of x_T, and receiving the output as the Hermitian of L. Matlab code for the LQ decomposition follows:

```
%lqHouse   Perform LQ decomposition of input matrix
function L = lqHouse(A)
% Find the size of the input matrix A.
[m, n] = size (A);

% Loop over each row of A.
for row = 1:m

% Compute the Householder vector v.
   x = 0;
   x(row:n, 1) = (A(row, row:n))';
   v = x;
   v(row) = v(row) + (x(row) / abs(x(row))) * norm(x);
   beta = - 2 / (v' * v);

% Apply the Householder reflection to the remainder of the
matrix.
   w(row:m, 1) = beta * A(row:m, row:n) * v(row:n);
   A(row:m, row:n) = A(row:m, row:n) + (w(row:m) * v(row:n)');
```

```
end  % for row

L = A;
```

Now the estimated noise covariance matrix can be expressed as:

$$\tilde{R} = (x_T)(x_T)^H = (LQ)(LQ)^H = LQ(Q^H L^H) = LL^H \quad , \tag{21}$$

and the Wiener-Hopf equation can be rewritten as:

$$w = \tilde{R}^{-1} v = (LL^H)^{-1} v \tag{22}$$

$$(LL^H)w = (LL^H)(LL^H)^{-1} v = v \tag{23}$$

Letting $t = L^H w$, $Lt = v$ can be solved for t by using forward-substitution since L is lower triangular. Then using $t = L^H w$, a solution for the adaptive filter vector, w, can be found by using back-substitution since L^H is upper triangular.

Now the added complication of the added diagonal loading factor in the adaptive beamforming stage is discussed. To solve this problem, define $x_T' = \left[x_T \; I\sqrt{\alpha}\right]$. Now $x_T' = L'Q'$, and the Wiener-Hopf equation, $w = ((L')(L')^H)^{-1} v$, can be solved as described above.

Finally, the denominator of equation (3) is a scalar which can be solved in the following manner. Using $\tilde{R} = (L')(L')^H$,

$$v^H \tilde{R}^{-1} v = v^H ((L')(L')^H)^{-1} v = v^H ((L')^H)^{-1} (L')^{-1} v \quad . \tag{24}$$

Letting $u = (L')^{-1} v$, $L'u = v$, which can be solved for u using forward-substitution, and $v^H \tilde{R}^{-1} v = u^H u$, which is a scalar.

ACRONYMS

CFAR – Constant False Alarm Rate

CPI – Coherent Processing Interval

FFT – Fast Fourier Transform

FIR – Finite Impulse Response

GMTI – Ground Moving Target Indicator

PCA – Polymorphous Computer Architecture

PRI – Pulse Repetition Interval

SNR – Signal-to-Noise Ratio

STAP – Space-Time Adaptive Processing

REFERENCES

[1] William G. Coate, "Preliminary Design Review: Kinematic Tracking for the PCA Integrated Radar-Tracking Application," MIT Lincoln Laboratory Project Report PCA-IRT-4, 25 February 2003, issued 6 February 2004.

[2] Gene H. Golub and Charles F. Van Loan, *Matrix computations*, 3rd Edition, Johns Hopkins University Press, 1996.

[3] Simon Haykin, *Adaptive Filter Theory*, 4th Edition, Prentice-Hall, Inc., 2002.

[4] Alan V. Oppenheim and Ronald Schafer. *Discrete-time signal processing*, Prentice-Hall, Inc., 1989.

[5] John G. Proakis, Charles M. Rader, Fuyun Ling, and Chrysostomos L. Nikias, *Advanced digital signal processing*. Macmillan Publishing Company, 1992.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE 27 February 2003	3. REPORT TYPE AND DATES COVERED Project Report

4. TITLE AND SUBTITLE

Preliminary Design Review: GMTI Narrowband for the Basic PCA Integrated Radar-Tracker Application

5. FUNDING NUMBERS

C — F19628-00-C-0002

6. AUTHOR(S)

A.I. Reuther

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Lincoln Laboratory, MIT
244 Wood Street
Lexington, MA 02420-9108

8. PERFORMING ORGANIZATION REPORT NUMBER

PR-PCA-IRT-3

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

DARPA/ITO
3701 Fairfax Drive
Arlington, VA 22203-1714

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

ESC-TR-2003-076

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

This report describes ground moving target indicator (GMTI) processing to be done for the PCA (polymorphism computing architecture) integrated radar-tracker application. GMTI processing of the raw radar data is done to extract targets; target reports are passed on to the tracker part of the integrated radar-tracker application.

The GMTI processing described in this report is used to minimize the effects of interference patterns and ground clutter and to recognize and extract targets on the ground. The radar is presumed to be using a narrowband signal. For each data set, adaptive processing is used to null interference patterns and ground clutter noise. These operations incur a high computational cost. Adaptive processing requires the calculation of an adaptive filter (involving a QR decomposition and forward/backsliding) at run time for each data cube, while target parameter estimation incurs a high computation cost as more targets are detected in each data cube.

14. SUBJECT TERMS			15. NUMBER OF PAGES 38
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Same as Report	19. SECURITY CLASSIFICATION OF ABSTRACT Same as Report	20. LIMITATION OF ABSTRACT Same as Report

12 February 2004

ERRATA

Document: Project Report PCA-IRT-3
Preliminary Design Review: GMTI Narrowband for the Basic PCA Integrated Radar-Tracker Application
27 February 2003, issued 6 February 2004

Please discard the copy of the above report recently sent to you and replace it with the enclosed copy.

Publications Office
MIT Lincoln Laboratory
244 Wood Street
Lexington, MA 02173-9108